Memory Usage Overview

There are several factors that can affect the amount of memory available for your job and the use of memory by your applications. We provide a number of tools you can use to monitor and manage memory usage.

Physical Memory Available on a Node When a Job Starts

The total physical memory of a Pleiades, Aitken, or Electra compute node varies from 32 gigabytes (GB) to 192 GB for Intel Xeon processors, or 512 GB for the AMD Rome processors. However, the amount of memory on a node that is available for a PBS job is less than the total physical memory, because the system kernel can use up to four GB of memory in each node.

When a job starts running, the PBS prologue tries to clean up the memory on the nodes used by the previous job that ran on the nodes, by flushing the job's data from memory to disk. If there is a delay in this cleanup attemptâ for example, due to Lustre issuesâ less memory will be available for the job. After the prologue has the opportunity to flush the memory, if the available memory is less than 85% of the total physical memory of the node, the prologue will terminate the job and return it to the queue. The node will be taken offline.

WARNING: For Aitken's Rome nodes, the PBS prologue will start your job if each of the assigned nodes has more than 85% of its 512-GB physical memory available. This could lead to a significant imbalance of available memory (from 435 GB to more than 500 GB) for nodes assigned to your job, and the possibility of running out of memory on nodes that had lower available memory when the job started.

Memory Used By and For Your Application

Segments and Processes

When you run an application, each of the following segments takes up space in the virtual memory:

text
: Executable instructions.

data
: Pre-defined dataâ global and static variables initialized by the programmer.

bss
: Uninitialized dataâ global and static variables that do not have explicit initialization in source code and/or are initialized to zero by the kernel.

stack
: Used for storing local variables and function parameters of a stack frame, which is created when a function is called and removed when the function returns.

heap
: Used for dynamic memory allocation during runtime.

shared libraries
: For example, math and MPI libraries.

You can run the Linux **`size`** command to list the bytes required (but not necessarily used) by the text, data, and bss memory segments.

Note: Some user applications have built-in reports of the memory requirement. These reports might not take into account the memory used by the shared libraries.

During runtime, the memory needed by each process in the application is paged (in 4 KB units) into the physical memory. The total amount of physical memory used by each process is called resident set size (RSS), which is reported in the RSS column of the **ps** command output or in the RES column of the **top** command output. The total physical memory used by application processes in a node can be estimated by the sum of the processes' RSS; keep in mind that the sum may be an overestimate, as the usage by the shared libraries might be counted more than once among processes.

Memory resource limits set by the system administrator for the Linux shells (such as **stacksize**, **memoryuse**, **vmemoryuse**, and so on) can affect your application. To see the settings, run the **limit** command (for csh) or **ulimit -a** (for bash) on a compute node. You might need to adjust one or more of the settings for your application to run properly.

Page Cache for Application I/O

Page cache, also known as buffer cache, is used to speed up the disk I/O of your application. Page cache usage is reported in the **Cache Mem** entry in the system usage section of the **top** command output.

If your job performs a large amount of I/O, there will be less memory available for your running processes. For some compute-intensive jobs, page cache management (limiting the amount of memory used by page cache) may improve performance. For more information, see Checking and Managing Page Cache Usage.

The /tmp Filesystem

The **/tmp** filesystem on a compute node is a fast, memory-based temporary local filesystem. You can access it directly via the **/tmp** path or by the **$TMPDIR** environment variable created by PBS for your job under **/tmp/pbs.jobid.pbspl1.nas.nasa.gov**. Usage by **/tmp** is included in the **Cache Mem** entry of the **top** command output.

Note: **/tmp** usage is limited to 50% of the total physical memory on the node. Using more than 50% results in the error **No space left on device**.

Tools for Checking Memory Usage

To help you assess the memory usage of your job, we provide the following NAS-developed tools:

qtop.pl
: Invokes the **top** command on the compute nodes of a job, and provides a snapshot of the amount of used and free memory of the whole node and the amount used by each running process.

qps
: Invokes the **ps** command on the compute nodes of a job, and provides a snapshot of the %mem used by its running processes.

qsh.pl
: Can be used to invoke the **cat /proc/meminfo** command on the compute nodes to provide a snapshot of the total and free memory in each node.

gm.x
: Provides the memory high-water mark for each process of your job when the job finishes.

vnuma

Provides memory usage on the node; total memory used by all user processes; memory used by each individual user process; and ratio of non-local memory access to local memory access.

The **vnuma** tool is accessible via **module load savors/2.x**. The other tools are installed in the **/u/scicon/tools/bin** directory.

Notes:

- The **qtop.pl**, **qps**, **qsh.pl**, and **vnuma** tools can be used only for jobs running on Pleiades, Aitken, and Electra. The **gm.x** tool can be used for jobs running on Pleiades, Aitken, Electra, and Endeavour.
- For Pleiades, Aitken, or Electra jobs, you can run the tools on the PFEs.
- If your Pleiades, Aitken, or Electra job uses more than one node, be aware that the memory usage reported in the PBS output file is not the total memory usage for your job; rather it is *the memory used in the first node* of your job.

If Your Job Runs Out of Memory

If your Pleiades, Aitken, or Electra job runs out of memory and is killed by the kernel, this event is likely recorded in system log files. See Checking if a PBS Job was Killed by the OOM Killer for instructions on how to check for messages in the log files. If the job needs more memory, see How to Get More Memory for Your Pleiades Job for possible approaches.

WARNING: For jobs running on Aitken's Rome nodes, the out-of-memory event is not currently recorded in the system log files and no email is sent to the owner of the job. If you suspect that your job may run out of memory, we recommend getting into the habit of monitoring memory usage using the tools listed above. If you need help, contact us at support@nas.nasa.gov.
For jobs running on Endeavour, the Linux kernel **cgroup** is used to enforce the resources allocated to your job, such as number of cores and amount of memory. When your job reaches the memory enforced by **cgroup**, your job will be terminated. If this happens, increase the memory request in your PBS script, and resubmit the job.

For additional information, see the recording and slides for the "Memory Usage on NAS Xeon-Based Systems" webinar, which are available in the HECC webinars archive.

Article ID: 216
Last updated: 17 Dec, 2021
Revision: 43
Running Jobs with PBS -> Optimizing/Troubleshooting -> Managing Memory -> Memory Usage Overview
https://www.nas.nasa.gov/hecc/support/kb/entry/216/